

EXHIBIT 5



Google

- [Cast](#)

Search

—

Language

[Home](#) [Guides](#) [Reference](#) [Samples](#) [Support](#)

—

Google

- [Cast](#)

- [Home](#)
- [Guides](#)
- [Reference](#)
- [Samples](#)
- [Support](#)

- Cast SDK

- [Get Started](#)

- [Registration](#)

- [Terms of Service](#)

- [Glossary](#)

- Sender Apps

- Develop Android Sender App

- [Setup](#)
 - [Integrate Cast](#)
 - [Customize UI](#)
 - [Automate UI Tests](#)
 - Add Advanced Features
 - [Media Tracks](#)
 - [Queueing](#)
 - [Intent to Join](#)
 - [Additional Features](#)

- [ExoPlayer Integration](#)

- Develop iOS Sender App

- [Setup](#)
 - [iOS Permissions Changes](#)
 - [Integrate Cast](#)
 - [Customize UI](#)
 - Add Advanced Features
 - [Media Tracks](#)

- [Queueing](#)
- [Additional Features](#)
- Develop Chrome Sender App
 - [Setup](#)
 - [Integrate Cast](#)
 - [Add Advanced Features](#)
- [Discovery Troubleshooting](#)
- [Guest Mode](#)
- Migrate Sender v2 App to CAF
 - [From Cast Companion Library](#)
 - [From Android SDK v2](#)
 - [From iOS SDK v2](#)
- Receiver Apps
- Develop Web Receiver App
 - [Overview](#)
 - [Styled Media Receiver](#)
 - [Create a Basic Receiver](#)
 - [Customize UI](#)
 - [Core Features](#)
 - [Streaming Protocols](#)
 - Add Advanced Features
 - [Tracks](#)
 - [Queueing](#)
 - [Ad Breaks](#)
 - [Live](#)
 - Debugging
 - [Chrome Remote Debugger](#)
 - [Cast Debug Logger](#)
 - [Command and Control \(CaC\) Tool](#)
 - [Error Codes](#)
- Develop Android TV Receiver App
 - [Overview](#)
 - [Core Features](#)
 - Add Advanced Features
 - [Tracks](#)
 - [Queueing](#)
 - [Ad Breaks](#)
 - Debugging
 - [Troubleshooting](#)
- [Migrate Receiver v2 to CAF](#)
- Design Guide
- [UX Guidelines](#)
- Design Checklist
 - [Overview](#)
 - [Cast basics](#)
 - [Cast button](#)
 - [Cast dialog](#)
 - [Cast autoplay](#)
 - [Sender app](#)
 - [Receiver app](#)
 - [Non-Touch](#)
 - [Touch](#)
 - [Changelog](#)
- Test Cases
- [Testing Cast Apps](#)
- Devices
- [Audio Devices](#)
- [Home](#)
- [Products](#)
- [Cast](#)
- [Guides](#)

Queueing

The Cast framework provides queueing classes that support the creation of lists of [MediaQueueItem](#) instances, which can be built from [MediaInfo](#) instances such as video or audio streams, to play sequentially on the receiver. This queue of content items can be edited, reordered, updated, and so forth.

Note: Review the [Google Cast Autoplay](#) for best practices when designing an autoplay/queueing experience for Cast.

The Receiver SDK maintains the queue and responds to operations on the queue as long as the queue has at least one item currently active (playing or paused). Senders can join the session and add items to the queue. The receiver maintains a session for queue items until the last item completes playback or the sender stops the playback and terminates the session, or until a sender loads a new queue on the receiver. The receiver does not maintain any information about terminated queues by default. Once the last item in the queue finishes, the media session ends and the queue vanishes.

Note: The [Styled Media Receiver](#) and [Default Media Receiver](#) do not support a queue of images; only a queue of audio or video streams is supported in the styled and default receivers.

Create and load media queue items

A media queue item is represented in the Cast framework as a [MediaQueueItem](#) instance. When you create a media queue item, if you are using the [Media Player Library](#) with adaptive content, you can set the preload time so that the player can begin buffering the media queue item before the item ahead of it in the queue finishes playing. Setting the item's autoplay attribute to true allows the receiver to play it automatically. For example, you can use a builder pattern to create your media queue item as follows:

```
MediaQueueItem queueItem = new MediaQueueItem.Builder(mediaInfo)
    .setAutoplay(true)
    .setPreloadTime(20)
    .build();
```

Load an array of media queue items in the queue by using the appropriate [queueLoad](#) method of [RemoteMediaClient](#).

Receive media queue status updates

When the receiver loads a media queue item, it assigns a unique ID to the item which persists for the duration of the session (and the life of the queue). Your app can learn the status of the queue in terms of which item is currently loaded (it might not be playing), loading, or preloaded. The [MediaStatus](#) class provides this status information:

- [getPreloadedItemId\(\)](#) method - If the next item has been preloaded, returns the preloaded item ID.
- [getLoadingItemId\(\)](#) method - Returns the item ID of the item that is currently loading (but isn't active in the queue) on the receiver.
- [getCurrentItemId\(\)](#) method - Returns the item ID of the item that was active in the queue (it might not be playing) at the time the media status change happened.
- [getQueueItems\(\)](#) (**Deprecated, use MediaQueue instead**) method - Returns the list of [MediaQueueItem](#) instances as an unmodifiable list.

Your app can also get the list of items using the [MediaQueue](#) class. The class is a sparse data model of the media queue. It keeps the list of item IDs in the queue, which is automatically synchronized with the receiver. [MediaQueue](#) doesn't keep all the [MediaQueueItem](#) because it will take too much memory when the queue is very long. Instead, it fetches the items on demand and keeps an [LruCache](#) of recently accessed items. You can use these methods to access the media queue:

- [getItemIds\(\)](#) method - Returns the list of all item IDs in order.